2016

Follow Me Robot

Victoria M. Edwards

Follow Me Robot

Victoria Edwards[1]

Abstract— **The idea of a personal robotic assistant, while once made up the tales of science fiction, is now moving closer and closer to reality. It is easy for two humans to walk together where one person knows the directions and the other person follows along. However, it is not a trivial problem for a robot to recognize a human, and follow side by side along with them. There are three key parts to this problem: 1.) Detection of a human, 2.) Tracking the human, and 3.) the robot's Motion strategy. Using baseline assumptions, I can simplify the detection process allowing this work to focus on tracking and the robot's motion strategy. In looking at tracking and motion, I will assume the existence of an ideal location for the robot to follow a human, and that it is possible for me to send the robot to that location. I will then discuss two different methods to handle motion: a strict control law approach, and a path planning approach. I also include discussion of experimental results of the two methods.**

I. INTRODUCTION

How a robot follows a human would at first appear to be a simple question. Humans follow objects around all the time with little to no effort, however, when that process is translated to a robot there are many questions that need to be considered. Consider even, a simplification of the problem to a robot which moves toward a stationary object. This still requires careful calculation of information received from the robot's sensors. I can break down this problem into three parts: 1.) Detection, 2.) Tracking, and 3.) Motion. Each of these components has been approached as its own respective field in robotics, and I will pull from the experience of the field to find a combination of algorithms that will give the robot the desired behavior: following a human.

A robot that can follow a human has an array of uses. For example, guided or assisted tours [1], interactive meetings[2], or a robot who works as a personal assistant. This type of set up could also be used to collect massive amounts of data on a persons day-to-day life, and the places the person goes. The robot could learn the person's habits, and over time be personalized to the human with which it has been paired. We are a long way off from an AI similar to that seen in *Star Wars'* BB-8, but a robot as a pet-like object is not an out of this world question within the next 50 years.

Object detection depends on the sophistication of the robot, and the type of data the robot is collecting. For example, a robot with a lidar scanner will have a different approach to detection than a robot with only a camera. Both forms of data, manipulated correctly, allow for the detection of any desired object. The added challenge is that this problem requires the robot to detect a human; not a uniform block of a fixed size and color. The difficulty is that humans come in a variety of shapes and sizes, and many approaches have been taken to find a solution to this question: [3], [4], [5], [6], [7], [8], [9], [10]. I decided to simplify the problem of detection by assuming the robots target human has some distinguishing pattern or color on their person. A variety of algorithms exist to pick out strong features in an image: [11], [12], [13]. I am using a color based strategy to pick out my desired object from the input image.

Fig. 1: This is Gollum, a Magellen robot with two wheel differential drive and a netbook

Since the 1980s a wide variety of tracking algorithms have emerged. State estimation is a way to try and predict where the robot's next position will be, based on previous positions of the human. Kalman filters [14], [15], particle filters [16], and optical flow [17] are three popular tracking algorithms that supply state estimation. After their initial introduction, a range of variations on these algorithms have been pursued by researchers.

Once I detect an object, and I have the ability to track it, the motion must be translated to the robot which at this point becomes a control theory question. There are many different kinds of control strategies, and in this work I implement a

*I would like to thank Professor Bruce Maxwell, Department Chair of the Colby College Computer Science Department, for the guidance, support, and mentoring over the past four years. I am especially thankful for his patience and time over the past year while working out all the problems I had making this project a reality.

[1] Victoria M. Edwards is with the Computer Science Department, Colby College, Waterville, Maine USA vedwards@colby.edu

proportional control law. This basic control law is attempting to minimize an error term to achieve the desired behavior for the robot.

The experimentation in this work was done on a Magellan Robot outfitted with a netbook. This is a two-wheel robot with differential drive and castor. We affectionately call the robot Gollum depicted in Figure 1.

Considering the three proposed questions I investigated a pure reactive control law and a state estimation control law. A reactive mechanism is one that responds to stimuli in the system, so the robot updates its position based on sensor readings it has received. A state estimation mechanism is a predictive model for where the robot should be based on a set of probabilities. These two control laws assumed there was an ideal space for a robot to be following a person.

The paper is organized as follows: in Section II we discuss related works, Section III outlines our assumptions and states the problem, Section IV goes over how we implement the control methods for a robot to follow a human, a discussion of the experimental setup is in Section V, and results from experiments are discussed in Section VI, Section VII is the conclusions and directions for future work.

II. Related Work

Following a human has peeked the interest of scientists in recent years. This is due to the growing potential of having a robot which can actually achieve and maintain following behavior over an extended amount of time. The problem can be broken into three key parts: Detection of the human, Tracking the human in the space, and devising a Motion strategy that ensures the robot will follow the target. Roboticists have taken several approaches that combine solutions from each individual task. The chosen methods are also limited by the sophistication and abilities of the actual robot.

To detect a human in a scene, the robot needs either a camera, a laser, a sonar system, or some combination of sensing methods. When considering camera data, the question about detecting a human in a space, enters the realm of computer vision. Object detection in an image has been popular since the early '90s. A human, however, comes in a wide variety of shapes and sizes, making it different from standard object detection. Recent papers looking to detect a human in an image consider solutions like: a more elaborate feature vector to maximize positive human detection [8], mosaicing techniques to segment the human shape [9], [10], using more sensors like LED lights, ultrasonic transponder and omnidirectional cameras [18], and others [3], [4], [5], [6], [7]. Work has also been done to track multiple objects in a scene [19], [20], [28], but for my question I want the focus of my detection to be on one singular object.

Once the human is detected, it is up to the robot to track and move as the human does. That means, even in an occluded environment, a robot needs to try its best to follow the human. The big question is: What is the most right thing that the robot should be doing? Many approaches have been taken to answering how a robot should move to follow a human. One approach to this question using UAVs instead of ground robots, is for the robot to follow the motion from the sensor readings in a headband worn by the operator of the robot [21].

Another spin off of this problem is seen in Kobilarov et al. [22], who proposed two different methods to follow a fast-moving person in an outdoor environment. The first method uses only visual input and an optic flow algorithm over the raw images to compute the egomotion. Then, they use a particle filter to determine the necessary state of the robot. The problem with this is that the measurements are not good enough to account for high variance in motion found in objects like trees. For their improvement on this method they takes in laser and camera data and use a Probabilistic Data Association Filter. This provides a single measurement for the robot state and in turn saw better results. The extra steps taken in these methods were necessary to handle the outdoor environment.

Focusing more specifically on the complete problem of detection, tracking, and motion, many researchers have tried a variety of approaches: [9], [4], [10], [23], [24], [25], [22]. More specifically, Chen and Birchfield [26] used Lucas-Kanade tracking, and flipped between tracking and detection states using a RANSAC-based procedure. Satake and Miura [27] use stereo based detection which allow for depth templates to be used to identify the human, and an extended Kalman filter is used for tracking. In [29] leg detection is done, and has a library of predefined orientations of legs. In this work tracking is done with a laser. The motion is defined by a human augmented mapping scheme; where the human guides the robot through the space, and then uses ideas from SLAM to generate a map.

Simaltaneous Localization and Mapping, SLAM, is a method in which the robot builds a map of its surroundings, and determines its place within that map. It is a more sophisticated version of state estimation which predicts where the robot falls within its unknown environment. As computing power increases many researchers are looking for the fastest and most effective way to get SLAM results, [30], [31].

Cosgun et al. [2] provide an algorithm which allows a robot to autonomously follow a human by using a cost and goal function. This method avoids the problem of having an undefined next step for the robot to move. To implement this algorithm, the authors built a tree of possible future positions, and do a restricted breadth first search to see the least costly step. They also consider an ideal location for the robot by applying costs to different motion patterns; i.e., moving from side to side.

Several works consider a pure path-pursuit following algorithm, [32], [3] which is a control mechanism based on a goal point defined for the robot. A computation is carried out to determine the curve between the robot and the goal point depending on the distance and orientation from the desired end result [33]. These papers are similar to the work I did in this project. I implemented a method that is inspired by a pure path-pursuit method. I used color histograms to detect a uniform color and size object and then tracked the object through the space. The robot respond to given goal points

depending on the detection of the object by a camera.

III. PROBLEM STATEMENT

Given a robot in an empty workspace, I will execute a detection algorithm over a visual image, and define a motion strategy for the robot to follow the path of the human as perceived by the visual data. Assume that an ideal location exists for the robot to be positioned when following the human, and let the robot be represented by the position vector $\dot{q} = [t_v, r_v]$, where t_v and r_v are the translational and rotational velocities. We will devise two control strategies that will manipulate $\dot{q}$ to execute the desired behavior. The horizontal coordinate of the detected object in the input image will be represented by B_x. Each strategy will propose solutions to the three questions within this problem: Detection, Tracking, and Motion planning. I limit the detection problem by looking for a unique colored object instead of trying to identify a human in general.

IV. METHODOLOGY

This control strategy is broken into three parts within the robot architecture. Detection and tracking is handled by the SVM module, motion is handled by the navigation module, and both pieces are tied together by a control state machine which sends and receives important information.

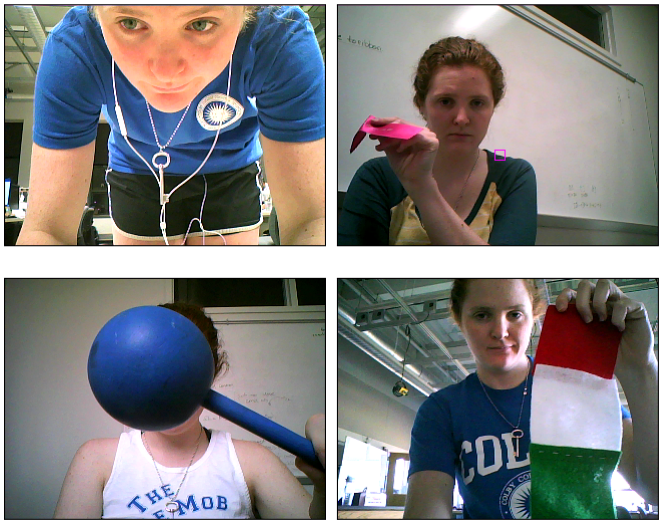

Fig. 2: These are examples of objects used for object detection. They help simplify the question about detecting a human. I used operators within the existing architecture, and found that the operator with the best result was the Italian Flag detector.

A. Reactive Control Strategy

To achieve detection I used the Italian Flag Operator; a pre-existing operator in the SVM architecture. Within the Italian Flag operator, a Kalman filter is implemented to identify the distinct color-pattern. This is handled by the SVM module, and when a message is requested it returns the last known position of the object to the robot.

The tracking strategy is to wait for the next input signal from the SVM module. That information is received by the control as B_x, where B_x is the horizontal position of the Italian Flag in the image, and it is then used below to update $\dot{q}$ such that the desired motion of the robot is achieved.

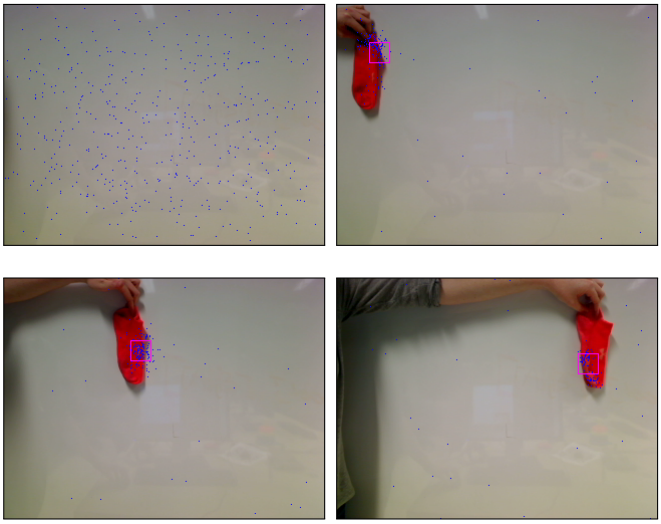

Fig. 3: This is the basic outline for a particle filter. The first image is a display of the random distribution of particles, the remaining three images depict the particles gathering at the orange sock.

B. State Estimation Strategy

State estimation is a predictive means to determine, based on previous steps, where the robot should be at the current moment in time. I implemented a particle filter which is outlined in algorithm 1. A particle filter is used over a Kalman filter because it is more robust while handling incorrect predictions for the robot's next position. A Kalman filter has limiting baseline assumptions that must be met to work properly. The biggest problem for this strategy is dealing with divergence cases where the robot has two directions to choose. If the robot chooses the wrong direction a Kalman filter has no way of going back and finding the object relative to the robot's new position. Thus a Kalman filter does not have the ability to look back in time, only forward, and a particle filter has the ability to look back in time to more accurately predict the position of the object.

A particle filter begins by first initializing N particles with random positions and equal weights. Then, for each time-step while the particle filter runs, the following occurs: the position of the particles is updated by a slight perturbation in its current location in the image. Next, new weights are computed. Any probability distribution can be used to compute weights for the particles. The probability distribution is used to show how accurate the prediction of the object's position is in the input image. I used a histogram to compute the necessary weights. At each particle, I took the input image color and compared it to the histogram. Particles that

were on a color with a high match in the histogram get a high probability, and particles with a low match get a low probability.

After updating the weights I execute a resampling algorithm to compute which particles should be kept based on the likelihood that they are near my desired object. To do this I start by considering the probabilities of each particle. The new list of particles is selected proportionally based on the old particles' probabilities. Particles with high probabilities are more likely to be picked over those with low probabilities. To do this in $\theta(N)$ time, I used an algorithm that first selects a random number between 0 and 1. Then I loop over the old particle list and sum the probabilities. For each particle I check to see if the current sum of probabilities is less than my preselected random number plus a step size. While this inequality is true I add that particle to my new particle list. Through this iterative method the particles will all begin to congregate around the desired object.

Finally, the estimated position of the object is computed by thresholding the particles, so only the particles with high enough probability are taken into account, and then the weighted average is taken. This value is sent to the control method as the predicted position of the desired object.

The particle filter handles the tracking and detection for this control strategy, and thus B_x, used in the motion strategy below, is constantly being updated by the prediction of the particle filter, and is not dependent on the detection of the object by the SVM module.

Algorithm 1 Particle Filter Algorithm

```
1: Initialize N Random Particles
2: while Particle Filter Runs do
3:     Update position of N particles
4:     Compute new weights of N particles
5:     Resample from the N particles
6:     Calculate the new estimated position
```

C. Motion

The motion strategy is the same for both of the above detection and tracking methods. They use a proportional control law which works to minimize an error term, θ_{error}.

Consider:

$$\theta_{error} = C_x - B_x, \tag{1}$$

where C_x is the x-coordinate for the center of the input image, and B_x is the x-coordinate for the location of the detected object in the image. This gives a horizontal error term which is needed for 2D translation and rotation on the ground plane. From here compute the translational and rotational velocities necessary for the robot to achieve its goal position:

$$r_v = \theta_{error} * k, \tag{2}$$

where k is a constant. Depending on the value of k, different behaviors can be achieved by the robot.

$$t_v = M_V * ((M_\theta - \theta_{error})/M_\theta), \tag{3}$$

where M_V is the maximum velocity the robot can travel forward, and M_θ is the maximum amount of rotation the robot can achieve. Again, both parameters can be manipulated to achieve different behaviors of the robot during motion. We update $\dot{q} = [t_v, r_v]$ with the newly computed t_v and r_v. This is passed to the navigation module to handle a move command, which computes the arc for the robot to travel along to match the desired rotational and translational velocity to achieve the predicted goal state.

D. Extra Manipulation To Control

To limit the robot moving in a completely wrong direction, due to latency in detections, additional elements were added to the control strategies. The first is a stopping criteria based on the change in time:

$$\Delta T = T_{current} - T_{lastdetection}, \tag{4}$$

where $T_{lastdetection}$ is the time at which the last input image was received.

$$S = \cos(((T_{max} - \Delta T)/T_{max}) * w), \tag{5}$$

where S is the stopping value. Since cosine has a range of $[0,1]$, it allows for detections with large time delays to have S close to 0, and for those with only a slight time delay, from the last detection, to be closer to 1. This allows for S to remain closer to 1 for larger ΔT, compared to a strictly linear computation which reduces the size of S too quickly. In this case $w \in [0, 1.57]$ as part of the domain of cosine.

The next criteria considered is when the robot is too close to the human. Without this term, the robot would continue moving toward the human with no regard for how close it gets to the human. So, I used sonar data to compute another stopping criteria, H:

$$H = (M_d - D_s)/M_d, \tag{6}$$

where M_d is the minimum distance the robot can achieve before it should stop or move in reverse, D_s is the average of the front three sonar values from the robot. This term will be negative, if it has passed the minimum distance the robot can get, and needs to move back to achieve the necessary stopping criteria.

Both S and H are multiplied to T_v to ensure the robot slows if either of these cases arise. Only S is applied to R_v because if the robot gets too close to something I want it to have the option to turn away from the object; depending on the next detection.

V. EXPERIMENTAL SETUP

The experiments were run on a Magellen Pro robot called Gollum, depicted in Figure 1. Gollum is a two-wheel differential drive robot with a castor. This means that the wheels move independently of one another, such that, when the robot needs to turn left the inside wheel turns backward and the

outside wheel turns forward, and vice versa for when the robot needs to turn right. The castor is a small stationary wheel attached to the front, which helps stabilize the robot when it slows down or speeds up. Gollum is controlled by a netbook computer, it connects to the robot through a USB to communicate navigation commands to the robot. These components set up the experimental platform that was used to test the methods discussed above.

The underlying software for Gollum is IPC message passing. At the simplest level, there is a control program that subscribes to navigation commands that come directly from Gollum, and publishes updated navigation commands for Gollum to execute. As the tasks the robot is trying to achieve become more complicated, more nodes can be added to communicate information through the system, i.e., a laser node.

I work in an obstacle free workspace 5.5x5.5m in size. In this space, I move freely with Gollum testing the strengths and limitations of my control strategies. I also ran experiments down a hallway to see how the performance changed with two walls on either side of the robot. To further my understanding of Gollum's motion, I calibrated 4 logitech cameras, and computed position coordinates for the robot and myself through the space; the results are discussed in greater detail in Section VI.

VI. RESULTS

A. *Qualitative Results*

The reactive control method achieved the desired result. I tried detecting many different objects as seen in Figure 2. I ultimately settled on the Italian flag Operator because the distinct color pattern limited false detections. Thus, Gollum was able to follow me through the space at a safe distance and speed. The stopping criteria worked as expected; it slows down as the time from the last detection grows. The motion, however, was discontinuous due to the latency between detections, and thus, Gollum displayed choppy motion. In an attempt to ensure the robot doesn't get too far off course from the human it is following, the robot will stop once enough time has passed from the last detection. So, there is no way around this time delay problem using the message-passing system. I could have gone directly into the navigation module, bypassing the control step, but ultimately I needed a better way to predict where Gollum should be without needing an exact detection.

The next step is to consider how to predict where the robot should be relative to the human without needing constant detection. Thus, I began looking at state estimation methods. The particle filter produces significantly smoother motion because it is constantly updating the position of the object. The dependency on the SVM module for detections is not as high because B_x is constantly receiving a new prediction from my particle filter. This method only sees discontinuities in motion when sharp turns are made, and when multiple objects of the same color are in the view. I was able to limit these factors in my quantitative results to see the overall improvement and continuity of motion.

Both methods were tested in the hallway, and both strategies worked the same as when tested in the lab. The reactive control law was prone to starting and stopping especially because the light in the hallway was not always consistent and thus made detecting the Italian flag challenging. The state estimation control law performed well and saw continuous motion, and because the hallway is straight this method did not experience many limitations to motion until the end when a sudden turn occurred to go back down the hallway.

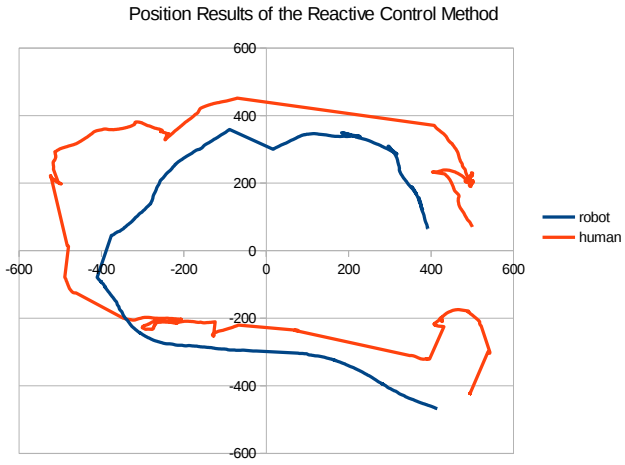

Fig. 4: Results from the reactive control law. Notice how the blue robot line and the orange human line are discontinuous and do not follow each other well. This is due to latency in detection, and the human having to move around to get the robot to see the Italian flag.

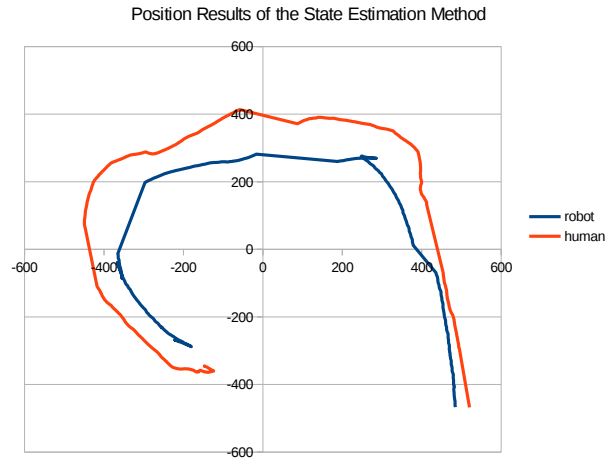

Fig. 5: Results from the state estimation control law. The lines are significantly smoother and the robot clearly follows along with the human.

B. *Quantitative Results*

To achieve these results, I set up a four camera camera system in the Davis Robotics lab. I ensured that there

were overlapping regions of all four cameras, calibrated the system, and then used Aruco tags to identify the human and the robot in the image. Using the computed calibrations, I was able to put the position of the robot in a global coordinate system, and achieve graphs and image results to show what the motion of the robot and the human looked like together.

These numerical results confirm the behavior I was describing above. It is clear in Figure 4 that the motion of the human, depicted in orange, is choppy, and clearly there is more motion of the human than the robot. Having been the test subject, I can attest to often losing Gollum and having to backtrack to have the robot looking at my desired target. This also highlights some of the difficulties this method suffered from due to varying light in the lab because I was using an operator dependent on color to detect the Italian flag.

In Figure 5, the blue and orange lines are much closer together and more continuous. There is less deviation from the path, and it is clear the overall motion was smoother. This is the desired behavior for the robot, once given the ability to predict what its next move should be without explicit detection from the SVM module.

Figure 6 shows the workspace with the calibrated camera system and the corresponding graph for the state estimation control law. The disconnections in the image lines are due to the orientation of the Aruco tag to the camera and when the netbook or my body are in the way of the camera meaning the tag cannot be seen. These results support the improved performance of the state estimation control law over the reactive control law.

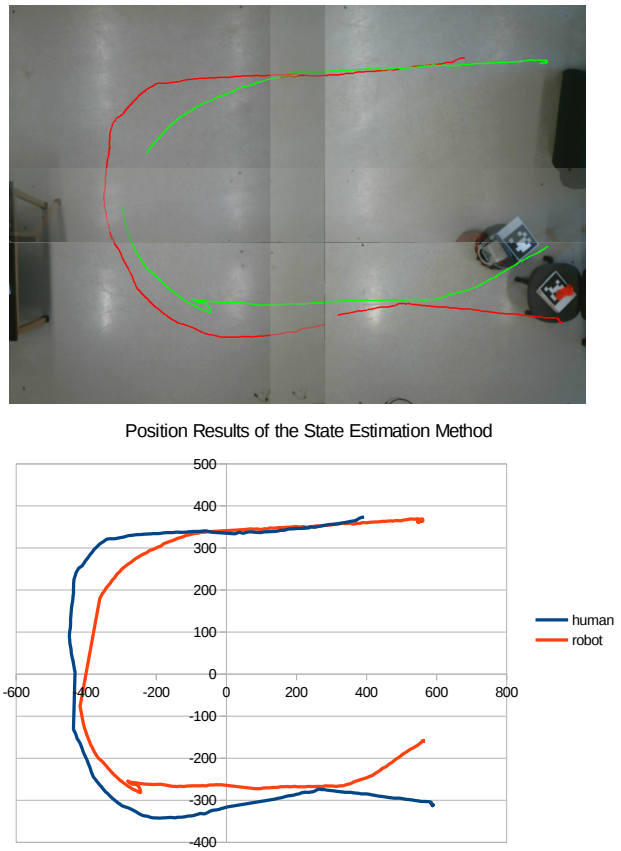

Fig. 6: Results from the state estimation control law on top of the workspace in the top image, where the green line is the robot and the red line is the human. Results as a graph in the bottom image.

VII. CONCLUSIONS

This project demonstrated two methods to have a robot follow a human. The first method was a reactive control law where a distinct color pattern was identified, and the robot updated its position depending on its current orientation in reference to the object it identified. In addition to that, dampening was applied to the motion to prevent the robot from getting too close to the human, and to reduce the incorrect motion when the amount of time from a detection was too large. The problem with this method is the latency in detection which made the motion discontinuous, and often the human lost the robot. The next method was a state estimation control law using a particle filter to determine where the human is in the scene, and allowing the robot to update its position based on the prediction of the particle filter. This method improved the smoothness of the robot, however, there is still no way for the robot to handle obstacles in the path between the human and the robot.

A direction for future work will be to determine a path estimation method that would allow the robot to follow a human around corners and through obstacles. A current consideration is to use a potential field method where the human will have an attraction point, and a repulsion point at all other obstacles. The challenge with this approach is the need for a map of the environment. Moving forward, more consideration would also need to be taken into account about what the specific use of the robot would be. If the robot was to follow many different people, it would need a way to identify which human to follow. However, if the robot was to follow only one specific human, it can be more customized to learn the human's routine.

REFERENCES

[1] A. Ustyuzhanin and D. Shepelev, "Toward Development of Reliable Mobile Robot Navigation System," *International Conference on Information Science and Control Engineering*, 2015.

[2] A. Cosgun, D. Florencio, and H. I. Christensen, "Autonomous Person Following for Telepresence Robots," *International Conference on Robotics and Automation*, 2013.

[3] G. Doisy, A. Jevtic, E. Lucet, and Y. Edan, "Adaptive Person-Following Alogrithm Based on Depth Images and Mapping," *IEEE/RSJ International Conference on Intelligent Robotis and Systems*, October 2012.

[4] T. Braun, K. Szentpetery, and K. Berns, "Detecting and Folloiwng Humans with a Mobile Robot," *EOS conference on industrial imaging and machine vision*, 2005.

[5] W. Ren and G. Li, "Person Re-identification Based on Relaxed Nonnegative Matrix Factorization with Regularitions," *International Conference on Pattern Recognition*, no. 22, 2014.

[6] R. Lan, Y. Zhou, Y. Tant, and P. C. L. Chen, "Person Reidentification Using Quaternionic Local Binary Pattern," *IEEE International Conference on Multimedia and Expo*, 2014.

[7] Y. Yoon, W. Yun, H. Yoon, and J. Kim, "Real-Time Visual Target Tracking in RGB-D Data for Person-Following Robots," *International Conference on Pattern Recognition*, no. 22nd, 2014.

[8] Y. Jiang and J. Ma, "Combination Features and Models for Human Detection," *CVPR IEEE International Conference on Computer Vision and Pattern Recognition*, 2015.

[9] A. Prati, F. Seghedoni, and R. Cucchiara, "Fast Dynamic Mosaicing and Person Following," *International Conference on Pattern Recognition*, 2006.

[10] M. Gupta, L. Behera, and V. K. Subramanian, "A Novel Approach of Human Motion Tracking with the Mobile Robotic Platform," *International Conference on Modelling and Simulation*, 2011.

[11] C. Harris and M. Stephens, "A Combined Corner and Edge Detector," *In Proceedings of the Fourth Alvey Vision Conference*, 1988.

[12] B. D. Lucas and T. Kanade, "An Iterative Image Registration Technique with an Application to Stereo Vision," *Proceedings 7th International Join Conference on Artifical Intelligence*, pp. 24–28, August 1981.

[13] D. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," 2003.

[14] R. Kalman, "A New Approach to Linear Filtering and Prediction Problems," *Transactions of the ASME-Journal of Basic Engineering*, 1960.

[15] G. Welch and G. Bishop, "An Introduction to the Kalman Filter," *UNC-Chapel Hill, TR 95-041*, July 2006.

[16] M. S. Arulampalam, S. Maskell, and N. Gordon, "A Tutorial on Particle Filters for Online Nonlinear/Non-Gaussian Bayesian Tracking," *IEEE Transactions on Signal Processiong*, 2002.

[17] B. Horn and B. G. Schunk, "Determining Oprical Flow," *ARTIFICIAL INTELLIGENCE*, 1981.

[18] A. Ohya, Y. Nagumo, and Y. Gibo, "Intelligent Escort Robot Moving Together with Human - Methods for Human Position Recotnition -," *SCIS ISIS*, pp. 275–278, 2002.

[19] A. Y. S. Chia, W. Huang, and L. Li, "Multiple Objects Tracking with Multiple Hypotheses Graph Representation," *Proceedings of the International Conference on Pattern Recognition*, 2006.

[20] A. Ohshima and S. Yuta, *Distributed Autonomous Robotic Systems 8*. Springer-Verlag, 2009, ch. Tracking and Following People and Robots in Crowded Environment by a Mobile Robot with SOKUIKI Sensor, pp. 575–584.

[21] K. Higuchi, K. Fujii, and J. Rekimoto, "Flying Head: A Head-Synchronization Mechanism for Flying Telepresence," *Artifical Reality and Telexistence*, 2013.

[22] M. Kobilarov, G. Sukhatme, J. Hyams, and B. Parag, "People Tracking and Following with Mobile Robt Using an Omnidirectional Camera and Laser," *Proceedings of the IEEE International Conference on Robotics and Automation*, October 2006.

[23] S. Shaker, J. Saade, and D. Asmar, "Fuzzy Inference-Based Person-Following Robot," *International Journal of Systems Applications, Engineering, and Development*, 2008.

[24] N. Pradhan, "Mobile Robot Navigation for Person Following in Indoor Environments," Ph.D. dissertation, Clemson University, 2013.

[25] T. Sonoura, T. Yoshimi, M. Nishiyama, H. Nakamoto, S. Tokura, and N. Matsuhira, "Person Following Robot with Vision-based and Sensor Fusion Tracking Algorithm," *IEEE/RSJ International Conference on Intelligent Robotis and Systems*, 2006.

[26] Z. Chen and S. T. Birchfield, "Person Following with a Mobile Robot Using Binocular Feature-Based Tracking," *IEEE/RSJ International Conference on Intelligent Robotis and Systems*, October 2007.

[27] J. Satake and J. Miura, "Robust Stereo-Based Person Detection and Tracking for a Person Following Robot," *IEEE International Conference on Robotics and Automation*, 2009.

[28] ——, "Stereo-Based Multi-Person Tracking using Overlapping Silhouette Templates," *International Conference on Pattern Recognition*, 2010.

[29] E. A. Topp and H. I. Christensen, "Tracking for Following and Passing Persons," *IEEE/RSJ International Conference on Intelligent Robotis and Systems*, pp. 2321– 2327, August 2005.

[30] G. Zhang and P. A. Vela, "Good Features to Track for Visual SLAM," *CVPR IEEE International Conference on Computer Vision and Pattern Recognition*, 2015.

[31] G. Bourmaud and R. Megret, "Robust Large Scale Monocular Visual SLAM," *CVPR IEEE International Conference on Computer Vision and Pattern Recognition*, 2015.

[32] R. Gockley, J. Forlizzi, and R. Simmons, "Natural Person-Following Behavior for Social Robots," *ACM/IEEE International Conference on Human-Robot Interaction*, 2007.

[33] R. C. Coulter, "Implementation of the Pure Pursuit Path Tracking Algorithm," *Technical Report CMU-RI-TR-92-01*, January 1992.